On the design of reliable and scalable networked systems

Hruby, T.

2016

document version
Publisher's PDF, also known as Version of record

Samenvatting

In dit proefschrift hebben we onderzocht of het mogelijk is om te profiteren van de nieuwste hardware ontwikkelingen, door het betrouwbare multiserver ontwerp op zo'n manier te veranderen dat het mogelijk wordt om er algemeen bruikbare besturingssystemen mee te bouwen, die zowel snel als betrouwbaar werken. Zo'n systeem maakt het gebruik prettiger voor de gemiddelde gebruiker, en verlaagt de kosten van het over-specificeren van hardware. Ook kan het de looptijd van langlopende wetenschappelijke verwerkingen verlagen. We hebben kunnen laten zien dat het in elk geval voor de netwerk systeem mogelijk is!

We bereiken ons doel door gebruik te maken van het feit dat multicore processoren meerdere delen van een multiserver besturingssysteem tegelijk kan uitvoeren, zonder daartussen te hoeven schakelen. Daardoor wordt het grootste snelheidsverlies van een multiserver systeem vermeden, zonder dat de structuur van het systeem veranderd hoeft te worden. Deze aanpak is echter geen simpele oplossing die zomaar werkt. Dit proefschrift onderzoekt de logische en praktische details van een betrouwbaar, snel en middel-bewust besturingssysteem, NEWTOS genaamd.

Allereerst hebben we laten zien dat communicatie tussen servers en drivers onpraktisch veel kost, en het zonder meer verdelen van deze componenten over meerdere cores het systeem niet sneller zal doen werken. We hebben puur user-space communicatie tussen paren van samenwerkende processen geïntroduceerd. De microkernel is niet meer nodig voor het uitwisselen van berichten, en is uitsluitend gebruikt om deze communicatiekanalen op te zetten. Deze kanalen zijn niet alleen vrij van microkernel tijdsverlies, maar deze nieuwe kanalen zijn ook asynchroon, dus communicerende processen hoeven niet op een rendez-vous te wachten, en kunnen doorwerken zonder te wachten op een beschikbare ontvanger.

Om de voordelen van zowel het nieuwe communicatie mechanisme als het verdelen van processen over beschikbare processor cores te demonstreren, hebben we

de netwerk systeem opnieuw gedaan. Dit is een voor snelheid en betrouwbaarheid een essentieel onderdeel van moderne besturingssystemen. We hebben gekozen voor het netwerk systeem omdat multi-gigabit verbindingen zo zwaar aan te sturen zijn dat er niet werd gedacht dat multiserver systemen hiertoe in staat zouden zijn. Dit in tegenstelling tot bij voorbeeld het opslagsysteem, waarbij de beperkende factor de opslagapparaten is. Alleen de net opkomende flash-gebaseerde techologieën kunnen vergelijkbare snelheid laten zien. Verdermeer zorgt de toenemende populariteit van netwerk file servers (NASsen) ervoor dat de netwerk systeem een belangrijk onderdeel aan het worden is van het opslagsysteem. Het nieuwe netwerksysteem van NEWTOS heeft de lat een enorm stuk hoger gelegd en het heeft laten zien dat een multiserver systeem, net als andere gangbare besturingssystemen, multi-gigabit snelheden kan verwerken.

Verder hebben we niet alleen de snelheid maar ook de betrouwbaarheid verbeterd. We hebben hetzelfde principe gebruikt als het verdelen van een monolithisch systeem in meerdere processen. We hebben de monolitische netwerk systeem in enkele onderdelen opgebroken, langs de snijlijnen van de al aanwezige protocol lagen. Dit maakt het her-opstarten na fouten simpeler, omdat de meeste onderdelen geen of weinig informatie hoeven te onthouden om hun werk te doen, wat het her-opstarten heel simpel maakt. Tegelijkertijd kunnen we methodes met verschillende snelheidskosten gebruiken om onderdelen te beschermen die wel veel moeten onthouden om hun werk te doen. Het grootste voordeel van deze opdeling is dat een fout in een makkelijk herstelbaar onderdeel geen effect heeft op de meer complexe onderdelen.

Systeemprocessen op vaste cpu kernen draaien beperkt uiteraard het aantal beschikbare kernen voor andere systeemprocessen, en natuurlijk bovenal voor toepassingen. Dit in tegenstelling tot hoe besturingssystemen worden begrepen te werken, gebaseerd op ervaring met monolithische systemen. Hoewel onze aanname is dat met het aantal toekomstig beschikbare kernen dit niet zozeer een beperking zal zijn, hebben we toch onderzocht wat de voordelen zijn van het draaien van een multiserver systeem op kernen waar meer draden dan kernen op beschikbaar zijn. Meer draden is makkelijker en goedkoper, en de systeemprocessen hebben meestal alleen een hardware houder nodig voor hun verwerkings context. Tegelijkertijd hebben we heterogene ontwerpen geëmuleerd om aan te tonen dat kleinere en simpelere kernen voldoende zijn om systeemprocessen uit te voeren, zelfs als het systeem zwaar belast wordt.

De beperking van onderdelen van de netwerksysteem op vaste kernen uit te voeren is dat een enkel onderdeel niet meer werk kan uitvoeren dan wat een enkele kern uit kan voeren. Ons netwerk systeem gebruikt weliswaar meerdere kernen, en kan dan ook meer werk aan dan een systeem dat maar op een enkele kern draait aan zou kunnen, wordt de schaalbaarheid beperkt door de meest veeleisende component. Maar we wilden ook meerdere software draden vermijden. Om de systeem toch op te schalen, hebben we een methode gevonden om meerdere exemplaren ervan uit te voeren. Elk exemplaar kan alleen werken, omdat ze strict geisoleerd worden en niet met elkaar hoeven te communiceren.

Omdat er meerdere onafhankelijke exemplaren van de netwerk systeem draaien, kan het multiserver systeem opschalen, en wordt de betrouwbaarheid op natuurlijke wijze vergroot, omdat een fout in een van de exemplaren netwerkverkeer niet onmogelijk maakt.

Tot slot hebben we laten zien dat de user-space communicatie die we in het netwerksysteem gebruiken om de onderdelen met elkaar te laten communiceren uitbreidbaar is, omdat we het ook gebruiken om de communicatie te verbeteren tussen het besturingssysteem en de toepassingen. Toepassingen kunnen dit uitgebreide mechanisme om diensten aan te vragen zonder dure system calls uit te voeren die het verloop van het proces onderbreken. We gebruiken dit model van user-space communicatie kanalen om netwerk sockets te bouwen. Verder hebben we aangetoond dat ouderwetsere event-gestuurde servers hier ook met behoud van eigen broncode van kunnen profiteren, die daardoor zowel in NEWTOS als in Linux beter kunnen presteren.

We hebben met succes aangetoond dat het mogelijk is om systemen te bouwen die even goed presteren als gangbare besturingssystemen, met bovendien een enorme voorsprong wat betrouwbaarheid betreft. Deze combinatie maakt ze uniek en zou ze op een breed toepassingsgebied welkom moeten maken.